AFRL-RI-RS-TR-2015-019

ADVANCED VISUALIZATION AND INTERACTIVE DISPLAY RAPID INNOVATION AND DISCOVERY EVALUATION RESEARCH PROGRAM TASK 8: SURVEY OF WEBGL GRAPHICS ENGINES

JANUARY 2015

INTERIM TECHNICAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■ **UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report was cleared for public release by the 88th ABW, Wright-Patterson AFB Public Affairs Office and is available to the general public, including foreign nationals. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RI-RS-TR-2015-019 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/ S /
ASHER SINCLAIR
Chief, Information Management
Technologies Branch

/ S /
JULIE BRICHACEK, Chief
Information Systems Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
JANUARY 2015	INTERIM TECHNICAL REPORT	APR 2014 – JUL 2014

4. TITLE AND SUBTITLE

ADVANCED VISUALIZATION AND INTERACTIVE DISPLAY RAPID INNOVATION AND DISCOVERY EVALUATION RESEARCH PROGRAM TASK 8: SURVEY OF WEBGL GRAPHICS ENGINES

5a. CONTRACT NUMBER
IN-HOUSE

5b. GRANT NUMBER
N/A

5c. PROGRAM ELEMENT NUMBER
62788F

6. AUTHOR(S)

Alexander C. Sarnacki III

5d. PROJECT NUMBER
PAVZ

5e. TASK NUMBER
IH

5f. WORK UNIT NUMBER
00

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory/RISB
525 Brooks Road
Rome NY 13441-4505

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory/RISB
525 Brooks Road
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RI

11. SPONSOR/MONITOR'S REPORT NUMBER
AFRL-RI-RS-TR-2015-019

12. DISTRIBUTION AVAILABILITY STATEMENT

Approved for Public Release; Distribution Unlimited. PA# 88ABW-2015-0170
Date Cleared: 16 JAN 2015

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Web browsers and portable devices open up many new application deployment possibilities for future software development, but they also have limitations that need to be addressed early. In order to ascertain if existing open source and commercial technologies would be sufficient to address current and future Air Force needs, it was desirable to complete a survey on existing web based Graphics Engines and associated products. In the end, nine (9) 3D engines, five (5) 2D engines, one (1) geospatial engine, and two (2) data driven document engines (these use data to control the creation and control of dynamic and interactive documents) were identified which are fruitful for further investigation.

15. SUBJECT TERMS
Graphics engines, WebGL, Open GL ES, 3D Graphics, 2D Graphics, Thin Client Graphics Engines, Web Browser Graphics Engines

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			**AARON W. MCVAY**
U	U	U	UU	30	19b. TELEPHONE NUMBER (*Include area code*) **N/A**

Table of Contents

1.0 Summary

Web browsers and portable devices open up many new application deployment possibilities for future software development, but they also have limitations that need to be addressed early. In order to ascertain if existing open source and commercial technologies would be sufficient to address current and future Air Force needs, it was desirable to complete a survey on existing web based Graphics Engines and associated products. In the end, nine (9) 3D engines, five (5) 2D engines, one (1) geospatial engine, and two (2) data driven document engines (these use data to control the creation and control of dynamic and interactive documents) were identified which are fruitful for further investigation.

2.0 Introduction

Nearly all software created has some form of visualization. This may be as simple as a Graphical User Interface (GUI) composed of buttons and menus, or complex such as a 3D visualization of aircraft in flight over a geospatially correct representation of terrain. The visualization component of any software application requires additional effort to create, and is not often the primary focus or goal of the application. The more effort, time, and money spent on creating the visualization, the less is available to develop the actual desired functionality.

Interactive webpages are written using two languages, the HyperText Markup Language (HTML) and JavaScript. JavaScript should not be confused with the Java programming language as the only commonality they have is in name. This was done as a marketing campaign to get people interested in adopting JavaScript. WebGL is a JavaScript Application Programmer's Interface (API) for rendering interactive 3D and 2D graphics within any compatible web browser without the use of plugins. WebGL is structured so that the control code is written in JavaScript and the graphics code is written in OpenGL shaders and executed on the graphics card hardware. A shader is a computer program that calculates rendering effects on graphics hardware and is usually directly loaded on to the graphics hardware. WebGL is a royalty free cross platform API that works by creating a 3D drawing context within HTML using the OpenGL shading language and GLSL ES (OpenGL for Embedded Systems). GLSL ES is designed for embedded systems like smart phones, tablets, and PDAs. WebGL can be cleanly combined with other web content that is layered on top of, or underneath, the 3D content.

To rapidly create visualizations an entire graphics engine is needed, not just access to a WebGL API. A graphics engine provides capabilities such as 3D model loaders, manipulation and moving of the viewpoint in the scene, manipulation and moving of collections of graphic primitives (lines, triangles, etc…) in the scene, addition and removal of collections of graphic primitives to the scene, and many more capabilities. Without a graphics engine providing these capabilities the software developer must recreate those difficult to perfect capabilities with every new software development effort. Recreating these capabilities for every new software development effort is time and cost prohibitive. Identifying

commercially available, open source, or GOTS, WebGL based graphics engines that provide all the required functionality would be very beneficial, as opposed to the other option, which is creating a new graphics engine from scratch. This work will greatly inform near-term future efforts on what decisions to make regarding a graphics engine and help steer the limited budget with respect to manpower onto the areas of opportunity with the greatest reward.

3.0 Methods, Assumptions, and Procedures

A search of the internet looking at web sites specializing in graphics, graphics engines, web browser applications, and games was conducted to identify potential graphics engines based on a variety of different factors. Many factors are taken into consideration when determining if a particular graphics engine should be considered for the next step of evaluation. Factors such as supporting hardware acceleration, having frequent and recent updates (an indicator that the product is being actively developed), thorough documentation, a plethora of examples and demonstrations, active discussion forums, advanced utilities such as camera controls, support for lights and materials, capabilities to load multiple file formats, support for custom code and shaders, the ability to collaborate with other engines such as physics engines, and more. The amount of detail for each engine identified is greatly varied based on the documentation which was readily available from each website and does not necessarily reflect the complete abilities of each engine. It can however be used as a relative measure of the maturity or intended audience for any particular graphics engine.

Graphics engines that exhibit many of these attributes should be considered for the next step of evaluation which would involve obtaining the graphics engine in question, installing, writing and implementing a variety of graphical tests, and evaluating the results.

Future evaluations should take into consideration the ease of all steps including obtaining and installing as both factors can be an issue that may impede work flow. However, emphasis should be put on the capabilities that each graphics engine provides.

4.0 Results and Discussion

The survey identified 52 graphics engines, 5 physics engines, and 6 DARPA XDATA related products. The following results are separated into several sections.

3D graphics engines. These are graphics engines that support 3D graphics and visualizations. Keep in mind that some 3D graphics engines have the potential to also do 2D graphics.

2D graphics engines. These are graphics engines that support 2D graphics and visualizations. Some of these graphics engines use WebGL for hardware acceleration for their 2D graphics.

Other graphics engines. These are graphics engines that provide graphics capabilities that are not easily defined as either 2D or 3D. For example, data driven document graphics engines.

3D physics engines. This survey was not intended to identify 3D physics engines so the only ones mentioned are ones that turned up while surveying the 3D graphics engines. These are engines that provide physics simulations, math, and related functions. They are used in conjunction with a separate 3D graphics engine as an add-on capability. Physics engines do not provide any graphical drawing or rendering functionality on their own.

2D physics engines. This survey was not intended to identify 2D physics engines so the only ones mentioned are ones that turned up while surveying the 2D graphics engines. These are engines that provide physics simulations, math, and related functions. They are used in conjunction with a separate 2D graphics engine as an add-on capability. Physics engines do not provide any graphical drawing or rendering functionality on their own.

DARPA XDATA Visualization related efforts and techniques. DARPA has an ongoing effort, XDATA, researching all aspects of extremely large data sets. One aspect of this research is the visualization of the data and results from large data analytics in a web-browser based environment. There were no specific graphics engines identified, but there were several techniques and applications that were worth mentioning.

The web addresses for each engine can be found in the Reference section of this document.

4.1. 3D Graphics Engines

4.1.1. Babylon.js

Babylon.js is a complete JavaScript framework and graphics engine for building 3D applications with HTML5 and WebGL. Babylon.js is a 3D engine based on WebGL and JavaScript. It has a complete scene graph with lights, cameras, materials, and meshes. It has a collision engine. It has a physics engine using Cannon.js. It features scene picking, anti-aliasing, animations, particle systems, sprites, and 2D layers. For optimization the engine supports utilizing frustum clipping, sub-mesh clipping, hardware scaling, selection octrees, offline mode (assets are saved locally to prevent reloading them), and incremental loading. It uses a standard material that is a per pixel material that supports diffuse lighting and texture, ambient lighting and texture, specular lighting, opacity texture, reflection texture (spheric, planar, cubic, and projection), mirror texture, emissive texture, specular texture, bump texture, custom materials, skybox, vertex color, four bones per vertex, and up to 4 lights (point, directional, spot, and hemispheric). It supports special effects including fog, alpha blending, alpha testing, billboarding, full screen mode, shadow and variance maps, rendering layers, post-processes (blur, refraction, black and white, fxaa, customs, and more) lens flares, and multi-views. It has texture support including render target textures, dynamic textures, video textures, compressed DDS textures, and 2D. It has camera support including arc rotate camera, free camera, touch camera, virtual joystick camera, and Oculus Rift camera. It supports meshes including mesh cloning, dynamic meshes, height maps, bones,

and constructive geometries. It supports importing and exporting different formats including OBJ, FBX, MXB, exporting Blender, exporting Cheetah3d, and support for drag and drop. Last update as of July 8, 2014 was on May 18, 2014.

4.1.2. C3DL

C3DL is a 3D library designed to simplify the use of WebGL. Named after Canvas 3D. It is an old product that is no longer developed or supported. Last update as of July 8, 2014 was on June 24, 2011.

4.1.3. Cesium

Cesium is a 3D JavaScript graphics library that specializes in creating 3D globes and 2D maps, in a web browser without a plugin, along with doing normal graphics. It was started by Analytical Graphics, Inc. (AGI), and is supported by an active open source community. It is a JavaScript virtual globe and map library. It uses WebGL for hardware accelerated graphics. It is cross platform, cross browser, and tuned for dynamic data visualization. It supports a 3D globe, 2D map, and a 2.5D Columbus View with transition between views with one line of code. It supports dynamic geospatial data visualization. It can animate dynamic scenes from CZML. It can visualize worldwide terrain from multiple sources and draw imagery layers using WMS, TMS, OpenStreetMaps, Bing, and ESRI standards. It can draw vector data from KML and ESRI shape files. It can draw polylines, polygons, circles, extents, billboards, ellipsoids, sensors, icons, labels, custom objects, and it uses a material system to change their appearance. It supports camera movement and flight paths. It uses low level geometric and rendering routines for performance. Each graphics layer can be alpha blended with the layers below it, and its brightness, contrast, gamma, hue, and saturation can be dynamically changed. It can draw the atmosphere, sun, stars, and water. It uses batching, culling, and JavaScript and GPU optimizations for performance. It implements precision handling for large view distances (avoiding z fighting) and large world coordinates (avoiding jitter). It supports individual object picking. It has camera support including spindle, flight, and free look. It supports materials and textures including water, images, and solid colors. It has procedural textures, classic texture modes, and Fabric which is a Javscript Object Notation (JSON) schema for describing and combining materials. For custom drawing it contains a thin abstraction over WebGL that provides built in GLSL uniforms for common transformations. It has built in GLSL functions for ellipsoids, rays, noise, and lighting. It supports shader programs and caching. It supports texture and cube maps. It has dynamic texture atlas packing. It supports buffers, vertex arrays, vertex layout, render states, frame buffers and render buffers. It supports geometric routines including view frustum and occlusion culling, vertex cache optimization, polygon triangulation and subdivision, tessellation, and bounding spheres and axis aligned bounding boxes. It has support for many math and geospatial functions including reference frames such as the World Geodetic System of 1984 (WGS84), International Celestial Reference Frame (ICRF), and east-north-up. It supports conversions such as longitude/latitude/height to Cartesian. It has types for cartesian, spherical, and cartographic positions as well as types for matrices and quaternions. It supports Catmull-Rom splines. It does Lagrange, Hermite, spherical linear,

and linear interpolation. It supports sun and moon positions. It supports Equidistant Cylindrical and Mercator 2D map projections. It supports computations on ellipsoids such as computing surface normal, circles, ellipses, and tangent planes. It has time, including Julian dates, leap seconds, time intervals, and UTC and TAI time standards. Last update as of July 8, 2014 was on June 2, 2014.

4.1.4. CloudMach

CloudMach is a HTML5 3D game engine. It works with WebGL for hardware acceleration. If a client does not support WebGL, CloudMach renders the 3D graphics in a cloud (server side rendering). CloudMach is a platform and hosting environment. Applications do not need their own servers or clusters. Last update as of July 8, 2014 was unclear.

4.1.5. CopperCube

CopperCube is a 3D authoring tool designed for non-programmers for creating 3D applications in a drag and drop fashion based on a behavior and action based logic system. It imports many 3D model formats. It has an integrated scripting api. It has a 3D authoring tool with deployment targets for Flash, WebGL, OpenGL ES, Windows, Mac, and Android. No programming is necessary to create 3D applications. It supports real time skeletal animation with no joint or weight limit and includes a simple animation editor. It features massive particle systems, billboards, skyboxes, 3D sounds, real lightmaps, real time normal maps, and adaptive precision to maintain high frames per second. To support user interfaces it has the ability to create 2D overlays which may contain buttons, text, and images. Last update as of July 8, 2014 was unclear.

4.1.6. CopperLicht

CopperLicht is a WebGL library and JavaScript 3D engine for creating 3D applications in the web browser. It uses the WebGL canvas supported by modern browsers and is able to render hardware accelerated 3D graphics without any plugins. It was originally intended to be used with CopperCube, but then changed to be its own product. It does 3D rendering based on a hierarchical scene graph. It supports many 3D model formats. It includes a full 3D world editor called CopperCube. It is optimized to render and animate huge 3D scenes. It supports skeletal animation and animated meshes with an unlimited number of joints and an unlimited number of weights. It compiles 3D meshes into binary files to reduce bandwidth and download quickly. It includes many pre-created materials and shaders. It implements fast pre-calculated lightmap support and dynamic light support. It supports particle systems, billboards, skyboxes, paths and splines, texture animation, and vertex color. It supports transparent objects with automatic item ordering to render correctly. It implements an animation and action system. It has an integrated 2D font and 2D primitive rendering system. It supports custom shaders and materials. It has a collision detection and response system. It supports picking and simple click detection. It has optimized, reusable 3D math classes all with collision test functionality. Last update as of July 8, 2014 was on April 18, 2014.

4.1.7. CubicVR.js

CubicVR.js is a 3D graphics engine that uses only your web browser, Javascript and WebGL (OpenGL ES 2.0 based) to produce high quality 3D graphics in real-time. Last update as of July 8, 2014 was on June 3, 2014.

4.1.8. Curve3D

Curve3D is a cross browser 3D engine written in JavaScript. It includes a high performance math library. Last update as of July 8, 2014 was on June 18, 2010.

4.1.9. Delight

Delight is a 3D engine designed to develop applications that run on any web enabled device or platform. Last update as of July 8, 2014 was unclear.

4.1.10. Enchant.js

Enchant.js. is a framework for developing simple 3D applications in HTML5 and JavaScript. It is open source and has extensive supporting documentation. It can be augmented with various plugins to increase functionality. It supports WebGL for 3D applications. It is object oriented, multiplatform, and event driven. It has an animation engine. It supports hybrid drawing with both the canvas API and DOM. Last update as of July 8, 2014 was on June 17, 2014.

4.1.11. GLGE

GLGE is a 3D JavaScript library intended to ease the use of WebGL, which is a native browser JavaScript API giving direct access to OpenGL ES2 allowing the use of hardware accelerated 2D/3D applications without having to download any plugins. It features keyframe animation, lights, normal mapping, animated materials, skeletal animation, Collada format support, parallax mapping, text rendering, fog, depth shadows, shader based picking, environment mapping, reflections/refractions, Collada animations, and portals. Last update as of July 8, 2014 was on June 20, 2014.

4.1.12. Goo Engine

The Goo engine is a 3D HTML5 and WebGL based graphics engine. It is modular so you may mix and match built in functionality to fit your needs. It is extensible so you may add your own functionality. It uses the entity-component-system philosophy which is a software pattern that encourages composition over inheritance. It consists of a Runner, which controls the application run loop, and a World which harbors all the Entities and provides a 3D coordinate system for them to exist in. An Entity is a basic object in the World which acts as a container and begins life as an empty container. An Entity can be given any set of functionality and features using Components. A Component is made for a specific purpose, such as a TransformComponent or a CameraComponent. There are many types of Components and each type gives an Entity its functionality. Entities can have multiple Components. Systems keep track of Components and run continuously to make sure Components are updated when needed. It has a state machine, scripting, and timeline to add interactivity to a scene. State machine components are added to Entities and let you

work with a large set of predefined actions. Last update as of July 8, 2014 was on July 1, 2014.

4.1.13. J3D

J3D is a WebGL based 3D engine. It consists of a JavaScript rendering engine and a scene exporter. Last update as of July 8, 2014 was on October 20, 2013.

4.1.14. JS3D

JS3D is a library for doing 3D graphics on a web page. Last update as of July 8, 2014 was on February 5, 2007.

4.1.15. Kick.js

KickJS is a 3D WebGL based graphics engine build for modern web browsers such as the most recent version of Chrome and Firefox. The source code is open source. It features Collada DAE and Wavefront OBJ models, built in shaders, a persistency model, view frustum culling, shadow maps, picking, render to texture, skybox, movie texture, directional, lights, point lights, ambient lights, event queue, keyboard and mouse input, serialization, and context lost handling. It has a GLSL shader editor, model tool, and extension viewer. Last update as of July 8, 2014 was on April 16, 2014.

4.1.16. Kuda

Kuda is a 3D JavaScript library and world editor that enables web developers to quickly build interactive 3D web solutions. It abstracts the complexity of 3D behaviors into easy to use building blocks for common functions, allowing developers to create compelling user experiences by setting up a complex sequence of events that respond to user input. It has a library of features such as advanced camera controls, transform manipulation, event sequencing, Heads Up Displays (HUDs), sprites, curve system, texture sets, doors, windows, pressure engine, and more. It has a browser based world editor which features a model browser, animation, camera viewpoints, particle effects, HUD, fog, messaging, and basic shapes. Last update as of July 8, 2014 was on April 14, 2012.

4.1.17. libGDX

libGDX is a cross platform open source 3D development library written in Java with some C/C++ for performance. It is based on OpenGL ES. libGDX allows the developer to write, test, and debug their application on a desk-top PC and use the same code on Android. It abstracts away the differences between a common Windows/Linux application and an Android application. It attempts to provide complete compatibility between desktop and mobile devices. It integrates with many 3rd party tools including Spine, Nextpeer, and Saikoa. It can do streaming music and sound effects playback for WAV, MP3, and OGG. It has abstractions for input for mouse, touchscreen, keyboard, accelerometer, and compass. It also has a gesture detector which detects taps, panning, flinging, and pinch zooming. For math it features matrix, vector, quaternion, bounding shapes, bounding volumes, frustum, Catmull-Rom splines, common interpolators, concave polygon triangulator, intersection testing, overlap testing, a wrapper for Box2D physics, a wrapper for Bullet physics, and more. For

graphics it features vertex arrays, vertex buffer objects, meshes, textures, framebuffer objects, shaders, immediate mode rendering emulation, simple shape rendering, automatic mipmap generation, and automatic handing of OpenGL ES context lost. For high level 2D APIs it features a custom bitmap manipulation library, orthographic camera, sprite batching, sprite caching, texture atlases, bitmap fonts, particle system, TMX tile map support, 2D scene graph API, and a 2D UI library. For high level 3D APIs it features a perspective camera, decal batching, billboards, particle systems, loaders for Wavefront OBJ, loaders for MD5, material system, lighting system, and has support for loading FBX models. It has a JSON writer and reader, with POJO support. It has a Xml reader and writer. It has a particle editor, texture packer, and bitmap font generator. Last update as of July 8, 2014 was on July 7, 2014.

4.1.18. O3D

O3D is an open source 3D JavaScript API created by Google for creating interactive 3D graphics applications that run in a web browser or in a XUL[1] desktop application. The new implementation of O3D is a JavaScript library implemented on top of WebGL. Last update as of July 8, 2014 was on October 25, 2011.

4.1.19. OSG.js

OSG.js is a 3D WebGL framework based on OpenSceneGraph concepts. It allows an individual to use an OpenSceneGraph like toolbox to interact with WebGL via JavaScript, and provides facilities for exporting various assets to the OSGJS format. Last update as of July 8, 2014 was on June 29, 2014.

4.1.20. PhiloGL

PhiloGL is a 3D WebGL framework for visualization applications, creative coding, and application development. It is built on JavaScript. It provides a rich module system covering program and shader management, IO, XHR, JSONP, Web Worker management, effects and tweening (the automatic addition of interpolated frames between specified animation key frames), and more. It is Open Source Software, licensed under the MIT license. Last update as of July 8, 2014 was on September 20, 2012.

4.1.21. PlayCanvas

PlayCanvas is a 3D application development platform that uses HTML5, JavaScript, and WebGL. It supports cloud based development so that you can edit applications collaboratively in real time with other developers on the web. It features a WebGL based 3D renderer, per pixel lighting, shadows, ambient lights, directional lights, point lights, spot lights, static and skinned meshes, post effects, audio, input, animation, skinned animation, entity system, and more. For physics it features rigid bodies, triggers, vehicles, joints, and more with full integration with the ammo.js physics engine. It does 3D positional audio via

[1] XUL (XML User Interface Language) is Mozilla's XML-based language for building user interfaces of applications like Firefox. The term XUL is sometimes used to refer to the whole Mozilla platform (e.g. XUL applications are applications using XUL and other components of the platform).

the WebAudio API. 3D models of many formats may be dragged and dropped into the PlayCanvas Designer. It has a scripting system. Last update as of July 8, 2014 was on July 11, 2014.

4.1.22. Pre3D

Pre3D is a 3D JavaScript rendering engine. It is a JavaScript library which projects a 3D scene into 2D and draws it into a canvas element. Last update as of July 8, 2014 was on January 26, 2011.

4.1.23. SceneJS

SceneJS is an extensible 3D WebGL based engine for high detail 3D visualization using JavaScript. It is based on a scene graph optimized for rendering large numbers of individually pickable and articulated objects. It features a scene graph with transform hierarchies, multiple scenes, and has auto GL context recovery. It has physics with rigid body dynamics and multiple physics systems. It supports frustum culling, level of detail, instancing, vertex sharing, and texture atlases. It has object picking and 3D ray picking. It has unlimited lights with ambient, directional, and point. It has camera support including orthographic, perspective, frustum, orbiting, and pick-fly-orbit. For geometry it supports triangles, lines, points, plane, box, cylinder, sphere, torus, terrain, vector text, vertex coloring, and automatic normals. Its texture support includes color, alpha, specular, bump, glow, reflection, multi texturing, texture animation, video texture, UV layers, and a texture plug-in API. It supports custom fragment shaders, custom vertex shaders, and transparency sorting. It supports skyboxes, height maps, and canvas image capture. Last update as of July 8, 2014 was on July 3, 2014.

4.1.24. SpiderGL

SpiderGL is a 3D JavaScript library for developing graphic applications for a web platform. It uses WebGL. It provides utilities, data structures, and algorithms to simplify WebGL development but still allows the integration of other WebGL code. SpiderGL provides typical structures and algorithms for real-time rendering to developers of 3D graphics web application, without forcing them to comply with some specific paradigm (i.e. it is not a scene graph), nor preventing low level access to the underlying WebGL graphics layer. Last update as of July 8, 2014 was on June 4, 2013.

4.1.25. Three.js

Three.js is a 3D WebGL graphics engine. The goal is to take complicated WebGL bindings and tack on an easy to use framework on top. It is a lightweight cross browser JavaScript library and API used to create 3D applications on a web browser. It features renderers for Canvas, SVG, and WebGL. It has effects such as anaglyph, cross-eyed, and parallax barrier. Scenes can add and remove objects at run-time and do fog. Cameras are supported with perspective and orthographic views along with input from trackball, FPS, and path. Animation is included with armatures, forward kinematics, inverse kinematics, morph, and keyframe. Several light types are supported including ambient, direction, point and spot

lights. It can cast and receive shadows. It supports materials with lambert, phong, smooth shading, and textures. It supports shaders with access to full OpenGL Shading Language capabilities including lens flare, depth pass, and extensive post-processing library. It has objects built with meshes, particles, sprites, lines, ribbons, bones, and more, all with level of detail. It has geometry such as plane, cube, sphere, torus, and 3D text. Geometry modifiers such as lathe, extrude, and tube. It includes data loaders for binary, image, JSON, and scene. It provides utilities such as full set of time and 3D math functions including frustum, matrix, and quaternion. It has export and import utilities to create Three.js compatible JSON files. Last update as of July 8, 2014 was on April 30, 2014.

4.1.26. Turbulenz

Turbulenz is a 3D HTML5 Javascript graphics engine. It is a modular 3D and 2D framework for making HTML5 based applications for web browsers and mobile devices. It is designed for performance, modularity, and customizability. It supports shaders with a simple shader based immediate mode API. It has vertex buffers, index buffers and textures that can be created, updated and destroyed dynamically. Multiple streams of vertex buffers can be used at the same time. It has support for 1D, 2D, 3D and Cube textures. It has asynchronous resource loading: multiple resource files can be downloaded on the fly and JavaScript code will be notified when the resource is available for usage. It has multiple image file formats including DDS, JPG, PNG and TGA. It has occlusion queries. It provides video playback support including WebM, and MP4. It can render video as texture. It has an easy-to-use efficient physics simulation for 2D and 3D. It supports rigid bodies and collision objects including plane, box, sphere, capsule, cylinder, cone, triangle mesh, and convex hull. It supports constraints such as point to point, hinge, cone twist, 6DOF, and slider. It has ray and convex sweep queries which return closest point of impact and surface normal. It has contact callbacks for collisions. It does sound supported with an easy-to-use efficient wrapper for hardware audio which features web audio, <Audio> tag, and OpenAL. It supports 3D sound sources with position, direction, velocity, gain, pitch, and loop. It supports asynchronous sound files loading so that multiple resource files can be downloaded on the fly and JavaScript code will be notified when resource is available for usage. It can decompress audio dynamically. It supports multiple sound file formats including OGG, WAV, and MP3. It supports query for platform capabilities so that it will load the best audio format for the platform. It has effect and filter support for features such as reverb, echo, and low pass. For input it has access to many input types including keyboard, mouse, Xbox360 pad, joysticks, wheels, touch, and multi-touch. Its high level scene graph approach implements a flexible JSON file format that can describe either a whole scene or individual meshes. It has asynchronous loading of external references so that if a scene contains references to external meshes they are all loaded in parallel and attached to the main scene when ready. It has support for optimal reuse of same mesh on different locations. It has a pluggable renderer system so that links between geometries, effects and materials are resolved at runtime. It can do an easy swap of multiple rendering techniques for same assets. It supports geometry sharing so that geometry information can be

optimally reused on multiple scene locations with different rendering effects. It does sorting and grouping so that visible nodes are sorted and grouped for optimal rendering, including opaque, transparent, and decal. It does 3D animation for scene geometry. It has skeleton/skinning animation. It supports animation controllers for interpolation, overloaded node, reference, transition, blend, mask, pose, skin, GPU skin, and skinned node. It can dynamically update scene data. It supports an unlimited number of lights including point, spot, directional, and ambient. It does texture based light falloff which allows multi-colored lights and cheap fake shadows. It supports materials with multiple texture maps such as specular color and intensity, normal vector, glow color, and alpha. It has pluggable post effects such as copy, fade-in, modulate, bicolor, and blend. It supports exponential shadow maps for reuse of texture shadow maps to save video memory. It does exponential depth information to avoid light bleeding. Last update as of July 8, 2014 was on June 26, 2014.

4.1.27. Unity

Unity is a 3D graphics engine and ecosystem to create 3D applications. It is fully integrated with a complete set of intuitive tools and rapid workflows to create interactive 2D and 3D applications. Unity's deferred lighting rendering uses the light pre-pass technique for the highest lighting and shadow fidelity for your application. Unity comes with 100 shaders ranging from the simplest (Diffuse, Glossy, etc.) to the very advanced (Self Illuminated, Bumped, Specular, etc.). Unity's shader system makes extensive use of fallbacks. Unity also supports graphics card emulation to simplify testing. It supports bypassing the rendering pipeline, and creating custom specific effects, with direct access to the Graphics and GL classes in Unity. The Graphics class is the raw interface to the drawing functions in Unity. Use it as a shortcut into the optimized mesh drawing functionality of Unity. The GL class is the low-level graphics library in Unity. Use this class to manipulate active transformation matrices and issue rendering commands similar to OpenGL's immediate mode. Unity developed an exclusive pre-computed visibility solution. Unity's occlusion culling reduces the number of rendered objects. Unity's iterative lightmap baking gives you complete control of your lightmapping workflow. The integrated lightmapping tool, Beast, bakes lights into textures for better performance. It provides dual lightmapping which uses one lightmap for distant scenery, and a second lightmap for only bounce light. On objects in the foreground, Unity fades in realtime direct lighting on top of the bounce lightmapped geometry. You may enhance lightmapped scenes with baked global illumination, sky lights, and emissive materials. You may use light probes to bake lighting onto moving characters and other dynamic objects and save on performance costs. For your Web, desktop and console applications, create high-quality, real-time shadows for directional, spot and point lights. Objects can cast shadows onto each other and onto parts of themselves. It supports render to texture. It provides a particle system tool which is a curve and gradient-driven modular particle system tool. Unity 4 brings you world collision functionality, bent normals, automatic culling, and external forces. Application audio is powered by FMOD which is a programming library and toolkit for the creation and playback of interactive audio. Unity can automatically import materials created in a 3D modeling package and set them up as

reusable assets. You can create your own materials in Unity, and choose from over 100 built-in shaders or write your own shader scripts. Unity features native support for procedural materials, powered by Allegorithmic's Substance technology. Procedural materials both reduce the size of your application and add impressive visual effects. It provides integrated 3D and 2D physics with soft bodies, rigid bodies, and a variety of joints. It has built in pathfinding with automatic navigation mesh (NavMesh) generation. NavMeshes describe the boundaries of any navigable space in your application and are used at runtime for path-finding. Last update as of July 8, 2014 was on May 15, 2014.

4.1.28. Voxel.js

Voxel.js is a collection of projects that make it easy to create 3D voxel applications in the browser. It is modular and requires WebGL. It has modules for physics, user interface, and multi-user. Last update as of July 8, 2014 was on March 15, 2014.

4.1.29. X3DOM

X3DOM is an open-source framework for 3D graphics on the Web. It can be freely used for non-commercial and commercial purposes, and is dual-licensed under MIT and GPL license. X3DOM tries to support the ongoing discussion in the Web3D and W3C communities about how an integration of HTML5 and declarative 3D content could look like, and it aims to fulfill the current HTML5 specification for declarative 3D content and allows including X3D elements as part of any HTML5 DOM tree. The goal here is to have a live X3D scene in your HTML DOM, which allows you to manipulate the 3D content by only adding, removing, or changing DOM elements. Last update as of July 8, 2014 was on May 8, 2013.

4.2. 2D Graphics Engines

4.2.1. CAAT (Canvas Advanced Animation Toolkit)

CAAT is a 2D director based scene graph renderer. It can render using Canvas, WebGL, and Cascaded Style Sheets (CSS). It is developed and maintained by the makers of CocoonJS. CAAT in essence is a multi-instance director-based Scene graph manager. It is multi-instance in the sense that you can set up an undefined number of directors for each web page. Each director is able to manage different Scenes with different timelines. The Scenes contain a graph of different Actor and Containers which conform the animation elements. It features seamless WebGL and CSS rendering engines. It has an unlimited number of Scenes per Director and a unlimited number of Actors and ActorContainers per Scene. It has hierarchically applied affine transformations (rotations, scales, and translations) and hierarchically applied alpha composition. It supports an unlimited number of timers per Scene. It has the ability to define complex paths for translations applied to Actors and ActorContainers. It has easing and interpolation functions for affine transformations and alpha transparency application. It has a homogeneous coordinate system, where each contained actor will receive correct input coordinates on its coordinate system regardless of its parents applied transformations. It has an abstracted input system, from mouse events to keyboard and accelerometer. It does resource management and preloading. Many types

of actors are supported, including sprites, shapes, text, path, and interpolator. It has an easily extendable framework and it performs properly in all major canvas enabled browsers. Last update as of July 8, 2014 was on July 2, 2013.

4.2.2. CakeJS

CakeJS is a 2D JavaScript scene graph library for the HTML5 canvas tag. It is a dynamic and extensible JavaScript scene graph. It has animation timelines and tweening. It supports picking and mouse events. It has node primitives (ImageNode, Rectangle, Circle, and more). It works well in complex layouts. It supports textures, picking, and sound. Last update as of July 8, 2014 was on Feb 9, 2012.

4.2.3. Canvas Engine

Canvas Game Engine is a 2D framework to do graphics in a HTML5 canvas. It supports sound, collisions, animation, and more. It supports scene structure for preloading, initialization, and rendering. It has preloading for images and sounds. It can display animations on a sprite. It supports the ability to create movement and dynamic transformations over time. It supports HTML5 audio API and SoundManager 2. It supports multi-touch gestures by implementing the Hammer.js library. It has many effects like element shaking. It supports collision detection. Last update as of July 8, 2014 was on April 11, 2014.

4.2.4. ChesterGL

ChesterGL is a WebGL 2D library, with canvas fallback, that focuses on ease of use and performance. It contains a simple scene graph. It has an extendable library. Some of its features include time based actions, simple scene graph, tiled map support, different shaders, and batched sprites. Last update as of July 8, 2014 was on July 28, 2013.

4.2.5. Cocos2D-JS

Cocos2D-JS is a port of Cocos2D and it is a multi-platform framework for building 2D applications. It is well maintained with large community. It is an open source 2D application framework released under the MIT license. Last update as of July 8, 2014 was on June 5, 2014.

4.2.6. Construct 2

Construct 2 is designed for rapidly creating 2D applications. It works at a very high level. You drag and drop objects into the scene and add behaviors to them. Construct 2 is an HTML5 application maker, meaning you are not actually writing JavaScript. Instead, you use actions, events and conditions to do the heavy lifting. It has a very active community with new releases weekly. The primary method of programming applications is through event sheets. Each sheet has a list of events which contain conditional statements or triggers. Once these conditions or triggers are met, actions or functions may be carried out. Events can be chained together using sub-events, allowing for complex behaviors. Events are created by selecting possible conditions and actions from an organized list. Behaviors work as pre-

packaged functions that you can assign to objects and reuse whenever needed. Last update as of July 8, 2014 was on April 24, 2014.

4.2.7. Crafty

Crafty is a 2D HTML5 JavaScript graphics engine from 2010. It supports collisions, sound, animations, and more. It uses an entity component system. It supports sprite maps. It uses an event system for custom event that can be triggered whenever on whatever. Last update as of July 8, 2014 was on January 31, 2014.

4.2.8. EaselJS

EaselJS is a 2D graphics engine inspired by Flash. It provides an API that is familiar to Flash developers, but embraces JavaScript sensibilities. It provides a display list to allow you to work with display elements on a canvas as nested objects. It provides a simple framework for providing shape based mouse interactions on elements in the display list. It is a JavaScript library designed to provide straightforward solutions for working with rich graphics and interactivity with HTML5 canvas. It consists of a full, hierarchical display list, a core interaction model, and helper classes to make working with 2D graphics much easier. It features images, vector graphics, animated bitmaps, simple text instances, containers that hold other DisplayObjects, and control HTML DOM elements. All display objects can be added to the stage as children or drawn to a canvas directly. All display objects on stage (except DOM elements) will dispatch events when interacted with using a mouse or touch. Last update as of July 8, 2014 was during the month of January 2014.

4.2.9. Frozen.js

FrozenJS is an open source 2D HTML5 graphics engine providing ease of use and rapid deployment through tooling and modularity. Last update as of July 8, 2014 was on September 14, 2013.

4.2.10. GameMaker

GameMaker is primarily for 2D graphics but allows the limited use of 3D graphics. It accommodates the creation of cross platform applications using a drag and drop interface, or a scripting language called Game Maker Language, which can be used to develop more advanced applications. It is intended to allow novice developers to create applications without have much programming knowledge. It is an older engine with roots back to 1999. In 2011 a HTML5 exporter was released so that applications developed can be exported to HTML5. It creates applications without having to write JavaScript. It has plugins and its own scripting language. It only supports the built-in “d3d” mesh format which is not compatible with other model formats. It supports fully integrated cross platform shaders, physics using the Box2D physics engine, audio, and networking. Last update as of July 8, 2014 was on May 29, 2014.

4.2.11. GC Devkit

GC Devkit is a 2D JavaScript graphics engine aimed at mobile devices. It is production ready with many applications currently developed. It is compatible with many current web browser development tools. Last update as of July 8, 2014 was on June 25, 2014.

4.2.12. ImpactJS

ImpactJS is a 2D JavaScript engine that allows you to develop 2D HTML5 applications. It is well used and well maintained with a large community. It is well documented. It includes a level editor and debug tools. Last update as of July 8, 2014 was unclear.

4.2.13. Isogenic Engine

Isogenic Engine is a 2D and isometric HTML5 graphics engine that supports real time multi user applications. It implements a scene graph based architecture. It is built in JavaScript. It provides many features such as particle emitters, tweening, cell based animation, and easily positioned text and fonts. It has a built in physics module that utilizes the Box2D physics engine. Last update as of July 8, 2014 was unclear.

4.2.14. LimeJS

LimeJS is a 2D HTML5 application framework for building fast, native-experience games for all modern touchscreens and desktop browsers. It has extra support for touchscreen devices. It is built with the Closure library, a JavaScript library built by Google. Last update as of July 8, 2014 was on June 1, 2014.

4.2.15. lychee.js

lycheeJS is a 2D Javascript library for development of HTML5 Canvas, WebGL, or OpenGL ES based applications. It is an environment independent JavaScript application engine. It ships with its own web server called Sorbet. Last update as of July 8, 2014 was on July 5, 2014.

4.2.16. melonJS

melonJS is a light weight 2D HTML5 engine. It features a 2D sprite based engine, standalone library, multiple audio channel support, basic physics and collision mechanisms, a basic set of object entities, basic vector math, tween effects, transition effects, object pooling, basic animation management, standard sprite sheet support, packed texture support, a state manager, system and bitmap fonts, mouse and touch screen support, mouse emulation on touch devices, built in support for CocoonJS, asynchronous messaging support, some basic GUI elements, and a customizable loader. Last update as of July 8, 2014 was on July 13, 2014.

4.2.17. MoSynch Mobile

The MoSync Mobile SDK is a 2D complete, rich, cross platform mobile application development SDK. It supports mobile applications. It is integrated with the Eclipse development environment. The framework produces native mobile applications for multiple platforms using C/C++, JavaScript, and HTML5. The MoSync platform can access parts of the native UI system on Android and iOS devices, and Windows Phone devices. The MoSync

NativeUI API has widgets for embedding webpages and OpenGL ES views in applications and all the UI widgets are handled from the same code base on both Android and iOS. It is also possible to run emulators from other SDKs, such as Android and iOS emulators ensuring that elements native to each OS has the right look and feel in their respective environments. Last production release as of July 8, 2014 was on May 30, 2013. There is a newer beta release.

4.2.18. Phaser

Phaser is based on Flixel and it is an open source 2D HTML5 graphics engine. It is maintained by Photon Storm. Development can be done using TypeScript or JavaScript. Uses WebGL if device supports it, otherwise reverts to an HTML5 Canvas. It uses Pixi.js internally for rendering. Phaser uses both a Canvas and WebGL renderer internally and can automatically swap between them based on browser support. This allows for fast rendering across Desktop and Mobile. Phaser supports shaders when running under WebGL. Phaser uses and contributes towards the Pixi.js library for rendering. It has a pre-loader for assets such as images, sounds, sprite sheets, tilemaps, JSON data, XML, and JavaScript files. It has a full body physics system with constraints, springs, and polygon support. It supports animation for sprites. It has a built in particle system. It has a built in camera system. It supports inputs from mouse, touch screen, multi touch, keyboard, and has functions to allow custom coded gesture recognition. Phaser supports both Web Audio and legacy HTML Audio. It automatically handles mobile device locking, easy Audio Sprite creation, looping, streaming and volume. With just a couple of lines of code Phaser can load, render, and collide with a tilemap. It supports CSV and Tiled map data formats with multiple tile layers and has tile manipulation functions such as swap tiles, replace them, delete them, add them, and update the map in real time. Phaser has a built-in Scale Manager which allows you to scale your game to fit any size screen. Control aspect ratios, minimum and maximum scales and full-screen support. Last update as of July 8, 2014 was on July 10, 2014.

4.2.19. Pixi.js

Pixi.js is a HTML5 2D rendering engine that uses WebGL for hardware acceleration for faster performance with standard canvas fallback if WebGL is not supported. It is built into Phaser. It works well with other engines that provide physics, sound, and other services. It supports multi touch. It uses a full scene graph approach. It features hardware acceleration using WebGL, a fast canvas renderer, a full scene graph, support for texture atlases, asset loader, sprite sheet loader, mouse and multi-touch interaction, text, bitmap font text, multi-line text, render texture, spine support, primitive drawing, masking, and filters. Last update as of July 8, 2014 was on July 11, 2014.

4.2.20. Quintus

Quintus is a 2D HTML5 engine. It is modular and lightweight with a concise JavaScript friendly syntax. It provides plugins, events, and a selector syntax. It provides a flexible component model in addition to traditional inheritance to make it easier to reuse functionality. Last update as of July 8, 2014 was on July 14, 2014.

4.2.21. Wade

Wade is a 2D graphics engine that targets all devices, phones, tablets, PCs, and consoles. It is a HTML5 framework with modular architecture. Last update as of July 8, 2014 was on July 1, 2014.

4.3. Other Graphics Engines

4.3.1. D3.js

D3.js is a JavaScript library for manipulating data into a visualization, or manipulating documents based on data. It allows you to bind arbitrary data to a data document model, and then apply data driven transformations to the document. Its emphasis on web standards gives you the full capabilities of modern browsers without tying yourself to a proprietary framework, while combining powerful visualization components and a data-driven approach to DOM (document object model) manipulation. It supports large datasets and dynamic behaviors for interaction and animation. Last update as of July 8, 2014 was on May 19, 2014.

4.3.2. DC.js

DC.js is a JavaScript charting library with native cross-filter support and allowing highly efficient exploration on large multi-dimensional datasets. It leverages the D3.js engine to render charts in CSS friendly SVG format. The rendered charts are naturally data driven and reactive therefore providing instant feedback on user interaction. It provides an easy yet powerful JavaScript library which can be utilized to perform data visualization and analysis in a browser as well as on a mobile device. Last update as of July 8, 2014 was on May 25, 2014.

4.4. 3D Physics Engines

4.4.1. Ammo.js

Ammo.js is a 3D physics engine. Ammo.js is a direct port of the Bullet physics engine to JavaScript, using Emscripten. The source code is translated directly to JavaScript so functionality should be identical to the original Bullet.js library. Ammo.js is free and open source with the zlib license. The Bullet physics library supports 2D and 3D physics operations and may be used to support other graphics engines. This does not render any graphics by itself. Last update as of July 8, 2014 was on May 16, 2014.

4.4.2. Bullet.js

Bullet.js is a 3D JavaScript physics engine. It is a JavaScript port of the Bullet physics engine. This does not render any graphics by itself. Last update as of July 8, 2014 was on July 2, 2011.

4.4.3. Cannon.js

Cannon.js is a 3D rigid body physics engine written in JavaScript. It is open source and written in JavaScript. It uses an iterative Gauss-Seidel solver to solve constraints. It is

lightweight and has a smaller file size than many other physics engine ports. It uses the SPOOK stepper. Last update as of July 8, 2014 was on August 11, 2013.

4.5. 2D Physics Engines

4.5.1. Box2DJS

Box2DJS is a 2D physics engine. Box2DJS is a JavaScript port of the Box2D Physics Engine. It was ported from Box2DFlashAS3. Its libraries have dependencies on each other and must be loaded in a specific order. This does not render any graphics by itself. Last update as of July 8, 2014 was on May 15, 2008.

4.5.2. Box2DWeb

Box2DWeb is a 2D physics engine. Box2dWeb is a javascript port of the Box2D Physics Engine. It was ported from Box2DFlash2.1a. It is similar to Box2DJS, but more recent, and without the library dependency problem. This port is stored in a single file. This does not render any graphics by itself. Last update as of July 8, 2014 was on June 15, 2011.

4.6. DARPA XDATA visualization related efforts and techniques

4.6.1. Bokeh

Bokeh is a Python interactive visualization library for large datasets that natively uses the latest web technologies. Its goal is to provide elegant, concise construction of novel graphics in the style of Protovis/D3, while delivering high performance interactivity over large data to thin clients.

4.6.2. Abstract Rendering

Abstract rendering is a technique that puts the emphasis on pixels instead of the shapes that the pixels are being used to create.

4.6.3. LineUp Web

LineUp Web is an interactive technique designed to create, visualize, and explore rankings of items based on a set of heterogeneous attributes that uses bar charts.

4.6.4. igraph

igraph is a library for creating and manipulating graphs. It is intended to be able to handle large graphs.

4.6.5. imMens

imMens is a system designed to support interactive visual exploration of large data sets with billions or more elements. The scalable visual representations are based on binned aggregation and support a variety of data types: ordinal, numeric, temporal and geographic. To achieve interactive brushing and linking between the visualizations, imMens pre-computes multivariate data projections and stores these as data tiles. The browser-based front-end dynamically loads appropriate data tiles and uses WebGL for data processing and rendering. In benchmarking tests, imMens is able to sustain 50 frames-per-second brushing

& linking among dozens of visualizations, with invariant performance on data sizes ranging from thousands to billions of records.

4.6.6. Lyra Visualization Design Environment

Lyra is an interactive environment that enables custom visualization design without writing any code. Graphical “marks” can be bound to data fields using property drop zones, dynamically positioned using connectors, and directly moved, rotated, and resized using handles. Lyra also provides a data pipeline interface for iterative visual specification of data transformations and layout algorithms. Lyra is more expressive than interactive systems like Tableau, allowing designers to create custom visualizations comparable to hand-coded visualizations built with D3 or Processing. These visualizations can then be easily published and reused on the Web.

5.0 Conclusions

There are several graphics engines that meet the criteria for further examination, analysis, and testing. Some of the factors taken into consideration are capabilities, maturity, documentation, examples, demos, discussion forums, developer support, and frequency of updates

Some of the graphics engines are designed for specific tasks. This survey conclusion sorts them into four (4) different broad categories. 3D graphics engines, 2D graphics engines, Geospatial 3D graphics engines, and Graph/Table/Document graphics engines.

Ideally the graphics engine that is chosen for any given effort should be based on the type of work that needs to be accomplished. There is potential for the use of multiple engines if the effort needs more capabilities than only one engine may provide. The ability for different graphics engines to collaborate, or draw into the same scene together, is another capability that should be explored and tested.

3D engines that are recommended for further study are:

1. Babylon.js
2. CopperLicht
3. Delight
4. Goo Engine
5. PlayCanvas
6. SceneJS
7. Three.js
8. Turbulenz
9. Unity

2D engines that are recommended for further study are:

1. GC Devkit
2. MoSynch Mobile
3. Phaser
4. Pixi.js
5. Wade

Geospatial 3D graphics engines. This is a unique category at the moment. There is only one identified graphics engine that specializes in geospatial visualization, the Cesium engine. This engine is capable of handling geospatial coordinate systems, different map types and projections, and addresses the problems that arise when using very large world coordinate systems. Leveraging these capabilities would be very beneficial in time, labor, and cost. It is also capable of doing general 3D graphics and these capabilities appear to be increasing. For efforts that require 3D graphics and some amount of geospatial visualization it is recommended to examine if Cesium is capable of meeting all of the requirements.

1. Cesium

Graph/Table/Document graphics engines that are recommended for further study are:

1. D3.js
2. Dc.js

6.0 References

Abstract Rendering. (n.d.). Retrieved from Abstract Rendering: http://www.darpa.mil/OpenCatalog/XDATA.html

AFRL Final Technical Report Guidelines - Federal Business Opportunities Opportunities. (2012, Jan 30). Retrieved Sept 30, 2013, from FedBizOpps.Gov: https://www.fbo.gov/index?s=opportunity&mode=form&id=9caca2212d07d1aa9b0363f363df55af&tab=core&_cview=0

Ammo.js. (n.d.). Retrieved from Ammo.js: https://github.com/kripken/ammo.js/

Babylon.js. (n.d.). Retrieved from Babylon.js: http://www.babylonjs.com/

Bokeh. (n.d.). Retrieved from Bokeh: http://www.darpa.mil/OpenCatalog/XDATA.html

Box2DJS. (n.d.). Retrieved from Box2DJS: http://box2d-js.sourceforge.net/

Box2DWeb. (n.d.). Retrieved from Box2DWeb: http://code.google.com/p/box2dweb/

Bullet.js. (n.d.). Retrieved from Bullet.js: https://github.com/adambom/bullet.js/

C3DL. (n.d.). Retrieved from C3DL: http://www.c3dl.org/

CAAT. (n.d.). Retrieved from CAAT: http://labs.hyperandroid.com/static/caat/

Cakejs. (n.d.). Retrieved from Cakejs: http://code.google.com/p/cakejs/

Cannon.js. (n.d.). Retrieved from Cannon.js: http://cannonjs.org/

Canvas Engine. (n.d.). Retrieved from Canvas Engine: http://canvasengine.net/

Cesium. (n.d.). Retrieved from Cesium: http://cesiumjs.org/

ChesterGL. (n.d.). Retrieved from ChesterGL: https://github.com/funkaster/ChesterGL/

CloudMach. (n.d.). Retrieved from CloudMach: http://cloudmach.com/

Cocos2d-JS. (n.d.). Retrieved from Cocos2d-JS: http://www.cocos2d-x.org/wiki/Cocos2d-JS

Construct 2. (n.d.). Retrieved from Construct 2: https://www.scirra.com/construct2

CopperCube. (n.d.). Retrieved from CopperCube: http://www.ambiera.com/coppercube/

CopperLicht. (n.d.). Retrieved from CopperLicht: http://www.ambiera.com/copperlicht/

Crafty. (n.d.). Retrieved from Crafty: http://craftyjs.com/

CubicVR.js. (n.d.). Retrieved from CubicVR.js: http://www.cubicvr.org/

Curve3D. (n.d.). Retrieved from Curve3D: https://github.com/sunetos/curve3d

D3.js. (n.d.). Retrieved from D3.js: http://d3js.org/

DC.js. (n.d.). Retrieved from DC.js: http://dc-js.github.io/dc.js/

Delight. (n.d.). Retrieved from Delight: http://delight-engine.com/#start

Easel.js. (n.d.). Retrieved from Easel.js: http://www.createjs.com/#!/EaselJS

Enchant.js. (n.d.). Retrieved from Enchant.js: http://enchantjs.com/

FrozenJS. (n.d.). Retrieved from FrozenJS: http://frozenjs.com/docs/

GameMaker. (n.d.). Retrieved from GameMaker: https://www.yoyogames.com/studio

GC Devkit. (n.d.). Retrieved from GC Devkit: http://docs.gameclosure.com/

GLGE. (n.d.). Retrieved from GLGE: http://www.glge.org/

Goo Engine. (n.d.). Retrieved from Goo Engine: https://www.goocreate.com/

igraph. (n.d.). Retrieved from igraph: http://www.darpa.mil/OpenCatalog/XDATA.html

ImMens. (n.d.). Retrieved from ImMens: http://www.darpa.mil/OpenCatalog/XDATA.html

ImpactJS. (n.d.). Retrieved from ImpactJS: http://impactjs.com/

Isogenic Engine. (n.d.). Retrieved from Isogenic Engine: http://www.isogenicengine.com/

J3D. (n.d.). Retrieved from J3D: http://creativejs.com/2011/07/j3d-javascript-3d-library/

JS3D. (n.d.). Retrieved from JS3D: http://wxs.ca/js3d/

Kick.js. (n.d.). Retrieved from Kick.js: http://www.kickjs.org/

Kuda. (n.d.). Retrieved from Kuda: http://code.google.com/p/kuda/

libGDX. (n.d.). Retrieved from libGDX: http://libgdx.badlogicgames.com/

LimeJS. (n.d.). Retrieved from LimeJS: http://www.limejs.com/

LineUp Web. (n.d.). Retrieved from LineUp Web: http://www.darpa.mil/OpenCatalog/XDATA.html

LycheeJS. (n.d.). Retrieved from LycheeJS: http://lycheejs.org/index.html

Lyra. (n.d.). Retrieved from Lyra: http://www.darpa.mil/OpenCatalog/XDATA.html

MelonJS. (n.d.). Retrieved from MelonJS: http://melonjs.org/

MoSynch Mobile. (n.d.). Retrieved from MoSynch Mobile: http://www.mosync.com/

O3D. (n.d.). Retrieved from O3D: http://code.google.com/p/o3d/

OSG.js. (n.d.). Retrieved from OSG.js: http://osgjs.org/

Phaser. (n.d.). Retrieved from Phaser: http://phaser.io/

PhiloGL. (n.d.). Retrieved from PhiloGL: http://www.senchalabs.org/philogl/

Pixi.js. (n.d.). Retrieved from Pixi.js: http://www.pixijs.com/

PlayCanvas. (n.d.). Retrieved from PlayCanvas: https://playcanvas.com/

Pre3D. (n.d.). Retrieved from Pre3D: http://deanm.github.io/pre3d/

Quintus. (n.d.). Retrieved from Quintus: http://html5quintus.com/

SceneJS. (n.d.). Retrieved from SceneJS: http://scenejs.org/

SpiderGL. (n.d.). Retrieved from SpiderGL: http://spidergl.org/

three.js. (n.d.). Retrieved from three.js: http://threejs.org/

Turbulenz. (n.d.). Retrieved from Turbulenz: https://turbulenz.com/

Unity. (n.d.). Retrieved from Unity: http://unity3d.com/

Voxel.js. (n.d.). Retrieved from Voxel.js: http://voxeljs.com/

Wade. (n.d.). Retrieved from Wade: http://www.clockworkchilli.com/index.php/developers

X3DOM. (n.d.). Retrieved from X3DOM: http://www.x3dom.org/

List of Symbols, Abbreviations, and Acronyms

API ……... Application Programmers Interface

C2 …… Command and Control

CZML ……. Cesium Language

DARPA …….. Defense Advanced Research Projects Agency

DOD …… Department of Defense

DOM …….. Document Object Model

GLSL ES …….. OpenGL Shading Language for Embedded Systems

GOTS …….. Government Off The Shelf

GUI …….. Graphical User Interface

HTML ……. HyperText Markup Language

JSON …… JavaScript Object Notation

KML …….. Keyhole Markup Language

PDA …….. Personal Data Assistant

TMS ……... Tile Map Service

WMS …… Web Map Service